SOBER Family of Stream Ciphers: A Review

Faheem Masoodi
Research Scholar
Department of Computer Science
AMU, Aligarh

Shadab Alam
Research Scholar
Department of Computer Science
AMU, Aligarh

M U Bokhari
Associate professor
Department of Computer Science
AMU, Aligarh

ABSTRACT
The SOBER family ciphers are generally constructed from a linear feedback shift register (LFSR), a non linear filter (NLF) and a form of irregular decimation called stuttering. Our work aims at having a comparative study of the SOBER family of stream ciphers with regard to design philosophies adapted and security aspects

General Terms
Algorithm, Design, Security

Keywords
SOBER, LFSR, NLF, Stuttering

1. INTRODUCTION
SOBER, a contrived acronym for seventeen octet byte enabled register, is a family of stream ciphers, widely used in embedded devices and was originally proposed by G. Rose in 1998 . The SOBER family ciphers include SOBER [1], SOBER-II [2], S16, S32 [3], SOBER-t8, SOBER-t16, SOBER-t32 [4] and SOBER-128[5].

SOBER [1] first member of the family, is a software stream cipher that was designed with the intent of meeting the essentials of embedded applications such as voice encryption in wireless telephones which place severe constraints on the amount of processing power, program space and memory available for software encryption algorithms.

Several variants of SOBER have been developed to strengthen its security or to be suitable for 16-bit and 32-bit processors. When various weaknesses were found in the original design of SOBER[6], it was superseded by SOBER-II and two SOBER variants: S16 and S32 [7], The analyses of SOBER-II found attacks that are specific to the overall structure of the cipher, rather than exploiting a weakness of the individual components used in the cipher.

S16 copies the structure of SOBER II[7] and is an enhancement of it to 16-bit wide arithmetic, while S32 is an extension of the design principles of SOBER to 32-bit wide arithmetic[4 ,8] and its structure is different from SOBER-II's and S16's. Nevertheless, there were opportunities for strengthening SOBER-II and S16 that could not be ignored. Consequently, three new versions based on 8-bit, 16-bit and 32-bit operations, called the t-class of SOBER ciphers [8] were developed as a substitute to SOBER-II, S16 and 32-bit version. The synchronous stream ciphers SOBER-t16 and SOBER-t32 were submitted to the NESSIE program [9]; as a stream cipher for 128-bit key and 256-bit key strength respectively. though both ciphers were found to fall short of the stringent NESSIE requirements but according to final NESSIE security report released in Feb 2003, there were only four stream cipher primitives which were considered during the phase II: BMGL[10],SNOW[11], SOBER-t-16 and SOBER-t-32[8].

Subsequent to NESSIE, SOBER-128 [5] is an enhanced version of SOBER-t32. The modifications directly address the concerns arising in the analyses of the t-class ciphers. The 128-bit key strength proposed for SOBER-128 is reduced from the 256-bit key strength proposed for SOBER-t32 to ensure that SOBER-128 provides far in excess of the stated security level. SOBER-128 was developed as a very conservative stream cipher, but with innovative (and flawed) message integrity functionality [5].

In the recent past enhancements to sober 128 have been developed in the form of NLS, NLSv2, SHANNON, TURING, and BOOLE.

2. DESCRIPTION
The SOBER families of ciphers are based on the same principle and have almost similar structure. Nearly all SOBER family ciphers consists of three basic components [5]. There is a Linear Feed-back Shift Register (LFSR), which uses a recursion formula to generate the stream of pseudo-random bits. Next a Non-Linear Function (NLF) combines these words in a non-linear way to produce the NLF-stream vn. Finally, the so-called stuttering produces the keystream zj by decimating the NLF-stream in an irregular fashion.

2.1 Linear Feedback Shift Register (LFSR)
The SOBER family cipher's LFSR's are typically based on recurrence relation over the Galois Field of order 2 GF (2w), W $\in$ {8, 16, and 32}, the output sequence (of bits) is defined by

$$S_{n+k} = C_{k-1}S_{n+k-1} + C_{k-2}S_{n+k-2} + \ldots + C_1S_{n+1} + C_0S_n, \text{ [4]}$$

Where Sn is the n-th output of the sequence, the constant coefficients ci , $0 \leq i \leq k-1$,are elements of GF(2) , that is, single bits, and k is called the order of the recurrence relation.
The LFSR is typically represented by the polynomial

$$P_1(x) = x^k + c_{k-1}x^{k-1} + c_{k-2}x^{k-2} + \ldots + c_1x + c_{0,} \text{ [4]}$$

where p1(x) indicates that multiplication and addition are over GF(21).
The length of the LFSR is 17, with every register containing one word, resulting in an internal memory of 544 bits. The contents

of LFSR at time t is called the state of the LFSR and is denoted by a vector S

$S_t = (s_t, s_{t+1}, s_{t+2}, \ldots s_{t+16}) = (r_0, r_1, r_2, \ldots r_{16})$

The next state of the LFSR is calculated by shifting the previous state one step, and calculating a new word st+17 as a linear combination of the words in the LFSR. The word st+17 is calculated as follows:

$$s_{t+17} = s_{t+15} \oplus s_{t+4} \oplus \alpha \cdot s_t ,$$

with α being a constant.

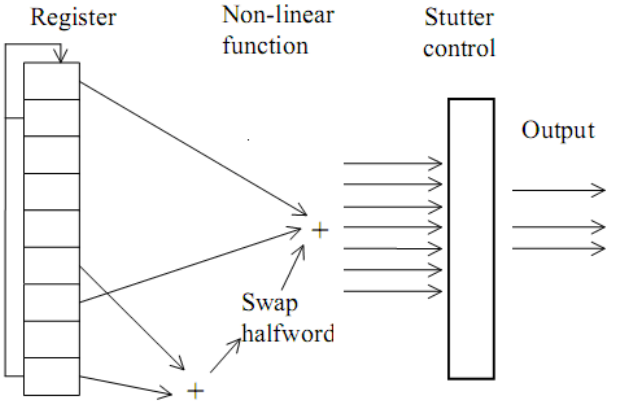

Figure 1: Block structure of SOBER family ciphers

2.2 The Non-Linear Function (NLF)

The non-linear function plays a vital role in strengthening the security of SOBER family by deterring the various attacks, that linear feedback and stuttering are susceptible to. The purpose of the NLF is to disguise the linearity in the LFSR stream. After every cycle of the LFSR, the NLF combines words from the register in a non-linear function; the outputs form the NLF stream vt.

The Non-Linear Function (NLF) at any instance of time t, combines five values denoted sn,sn+1,sn+6,sn+13,sn+16, from certain positions of LFSR state, using a technique involving the rotation of partial sums and calculates one output, called vt. This output can be written as:

vt =((f(st ⊞ st+16) ⊞ st+1 ⊞ st+6) ⊕ K) ⊞ st+13 [8]

In above equation, K is a key dependent constant used in t-class and succeeding ciphers of family and is derived immediately after the secret session key is loaded, ⊞ denotes addition modulo232 , ⊕ represents bitwise XOR and f is a non-linear function.

The function f serves three purposes [12]. Firstly, it removes the linearity in the least significant bit and adds significant non-linearity to the remaining bits. Secondly, it ensures that the addition of r0 and r16 does not commute with the addition of r1 and r6. Thirdly it ensures that every bit of the output of the NLF depends on every bit of r0 and r16. XORing the value of Konst into the NLF has two purposes. One, it increases the complexity of any attack (excluding exhaustive key search) as there are now 216 possible NLF functions. And two, the Konst is XORed (rather than added) so as to lower the probability of the addition of r13 commuting with the addition of r1 and r6. There is still a small probability that the operations will commute, but this probability is low and relies on the value of Konst.

The interior design of the function f used in case of SOBER-t16 and SOBER-t32 is pictured in Figure 2[4]. First the input is partitioned into a high part containing the 8 most significant bits, and a low part containing the remaining bits. The high part addresses an SBOX with W bits of output. The 8 most significant bits are directly taken as the f-function output, whereas the least significant part of the SBOX output is first XORed to the low part from the input

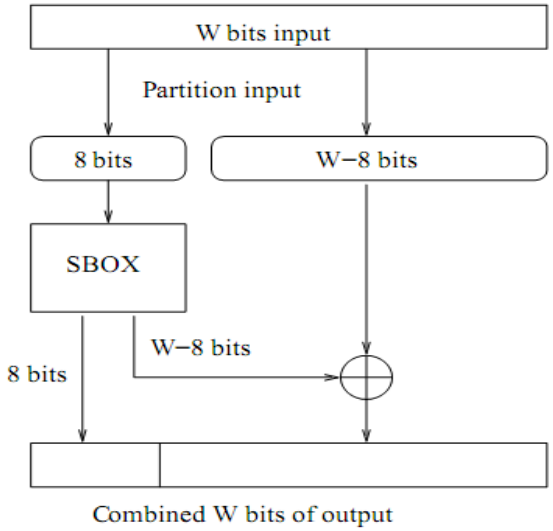

Figure 2: The structure of the *f*-function in SOBER-t16 and SOBER-t32

2.1 Stuttering

Stuttering has been considered as the most ad-hoc part of the design of the SOBER family ciphers[4], but still appears in almost every variant of SOBER except SOBER 128 and offers the best increase in security. It introduces irregular decimation in the output produced by the NLF to avoid the reconstruction of the state particularly by correlation attack. In these implementations, occasional words of nonlinear output are used to determine the inclusion of other words in the output stream. When the generator is started, the first output word is taken to be used as a stutter control word (SCW)[13]. Each stutter control byte is broken into pairs of bits called dibits, with the least significant pair of bits being used first. The dibits provide the cipher with instructions regarding how many times to cycle the LFSR, whether to output an NLF output, and how this value is included in the key stream. When all dibits in the SCW have been used, the LFSR is clocked once and a new SCW is read from the output of the NLF.

There are four possible values for each of the two-bit stutter controls[3]. These determine the output from the stream generator as follows:

(0, 0) The register is cycled but no output is produced.

(0, 1) The register is cycled, and the nonlinear output XOR the constant C becomes the output of the generator. The register is then cycled again.

(1, 0) The register is cycled twice, and the (second) nonlinear output becomes the output of the generator.

(1, 1) The register is cycled, and the nonlinear output XORed with the complement of constant C becomes the output of the generator.

3. COMPARATIVE ANALYSES

In this section we have carried out an analyses based on the design features like keying the stream cipher, byte order considerations, memory requirements, key strength, polynomials used, LFSR over GF (2^w), and security issues.

3.1 Keying the Stream Cipher

3.1.1 Byte Order Considerations

SOBER,SOBER II,S16,S32,SOBER t-8,t-16 and t-32 utilize native processor operations on integer data items, but are expected to accept keys which are simply strings of bytes, and to produce a stream of bytes as output for encryption purposes.[3]. To ensure compatibility between implementations running on different processors, a translation between native byte ordering and external byte ordering is necessary. Since all internet standards are defined using "big-endian" byte ordering[14], in which the most significant byte of a multi-byte quantity appears first in memory, this is what is chosen for SOBER,SOBER II,S16,S32,SOBER t-8,16.

On "little-endian" machines, the key and the words of the output stream must be byte reversed before being loaded or XORed into the buffer, respectively

In contrast SOBER-128 is entirely based on 32-bit word operations internally [5], but the external interface is specified in terms of arrays of bytes. Conversion between 4-byte chunks and 32-bit words is done in "little-endian" fashion irrespective of the byte ordering of the underlying machine. This is a break with the tradition of previous members of the SOBER family

3.1.2 Key Loading

The SOBER family ciphers are designed for applications in wireless telephony. Loss of synchronization haunts such applications a great deal. In order to avert such calamitous loss, one solution adopted by GSM system is to have each encrypted frame implicitly numbered with a frame key, and the stream cipher re-keyed for each frame with the secret session key and the frame key. SOBER supports such a two tier keying structure [1]. S16 shares this two-level keying structure. S32 was not designed to support such a two tier keying structure[3]; All t-class ciphers and SOBER 128 have been designed to support the two-tiered keying structure in addition to the standard mode of operation[5]. The cipher is keyed and re-keyed by using operations that transform the values in the register under the influence of key material. Two principle operations are employed:

Include(X): adds the word X to r_{15} modulo 2^w;w={8,16,32}.

Diffuse(): cycles the register and this operation clocks the register, obtains the value by XORing the output of the NLF with r_4. Table 1 below gives a comparison of key length, bit operation and memory requirements in bytes of different variants of SOBER family

Table 1: The Key length, bit operations and memory requirements of SOBER family ciphers

Cipher	w-bit Operation	Max Key Length (bits)	Memory (bytes)
SOBER	8	128	54
S-16	16	194	108
S-32	32	256	78
SOBER t8	8	64	54
SOBER t16	16	128	105
SOBER t32	32	256	210
SOBER 128	32	128	140

3.2 Linear Feedback Shift Register over GF (2w)

The LFSRs of SOBER, SOBER-II, SOBER-t8 are over GF (2^8); The LFSRs of S16, SOBER-t16 are over GF (2^{16})[3]. The LFSRs of S32, SOBER-t32, and SOBER-128 are over GF (2^{32}). The length of S32 is 9 stages and the lengths of the others are all 17 stages. Table 2 below compares the irreducible polynomials used by SOBER family ciphers and their hexadecimal representation

4. CONCLUSION

SOBER family of ciphers came into being to replete the void of unavailability of a standard in embedded applications such as wireless telephony. In this review we analyzed different components of SOBER family ciphers and carried out the comparative study of various variants of SOBER in terms of key loading, byte ordering, memory requirements, polynomials employed and implementation of LFSR. Out of the clump of ciphers discussed in this study, SOBER-128 outplayed all other ciphers and remains unbroken as a stream cipher but the Message Authentication Capability (MAC) introduced in this cipher was trivially broken. Subsequently improvements have been made in the succeeding releases of SOBER-128 like NLSV2, TURING, SHANNON and BOOLE

Table 2: The irreducible polynomial used in SOBER family ciphers

Cipher	Irreducible Polynomial	Hexadecimal Representation
SOBER	$X^8+X^6+X^3+X^2+1$	**0x14D**
S-16	$X^{16}+X^{14}+X^{12}+X^7+X^6+X^4+X^2+X+1$	**0x150D7**
S-32	$X^{32}+(X^{24}+X^{16}+X^8+1)(X^6+X^5+X^2+1)$	**0x165656565**
SOBER t8	$X^8+X^6+X^3+X^2+1$	**0x14D**
SOBER t16	$X^{16}+X^{14}+X^{12}+X^7+X^6+X^4+X^2+X+1$	**0x150D7**
SOBER t32	$X^{32}+(X^{24}+X^{16}+X^8+1)(X^6+X^5+X^2+1)$	**0x165656565**
SOBER 128	$x^{17}+X^{15}+X^4+\alpha$	**0x170D**

Table 3: The feedback function of SOBER family ciphers

Cipher	Feedback Function
SOBER	$S_{n+17}=206\otimes s_{n+15}\oplus s_{n+4}\oplus 99\otimes s_n$
S-16	$S_{n+17}=E38216\otimes s_{n+15}\oplus s_{n+4}\oplus 673C16\otimes s_n$
S-32	$S_{n+9}=6327DFDA16\oplus s_{n+4}\oplus 2\otimes s_n$
SOBER t8	$S_{n+17}=CE16\otimes s_{n+15}\oplus s_{n+4}\oplus 6316\otimes s_n$
SOBER t16	$S_{n+17}=E38216\otimes s_{n+15}\oplus s_{n+4}\oplus 673C16\otimes s_n$
SOBER t32	$S_{n+17}=s_{n+15}\otimes s_{n+4}\oplus C2DB2AA316\otimes s_n$
SOBER 128	$Sn+17=s_{n+15}\oplus s_{n+4}\oplus\alpha\otimes s_n$

5. REFERENCES

[1]. G. Rose, "SOBER: A Stream Cipher based on Linear Feedback over GF (2^8)". Unpublished report, QUALCOMM Australia, Suite 410 Birkenhead Point, Drummoyne NSW 2047 Australia, 1998.See http://people.qualcomm.com/ggr.QC

[2]. G. Rose, "SOBER II: A Fast Stream Cipher based on Linear Feedback over GF(2^{32})".Unpublished report, QUALCOMM Australia, Suite 410 Birkenhead Point, Drummoyne NSW 2047 Australia, 1998. See http://www.home.aone.net.au/qualcomm

[3] . Greg Rose, "S16 &S32: Fast Stream Cipher based on Linear Feedback over GF (2^n)".-2000

[4] G. Rose and P. Hawkes, "The t-class of SOBER stream ciphers", unpublished report, QUALCOM Australia, 1998. See http://www.home.aone.net.au/qualcomm.

[5] . Philip Hawkes and G. Rose ,"Primitive Specifications of SOBER 128," IACR ePrint Archive, http://eprint.iacr.org/2003/81/, 2003.

[6] D. Bleichenbacher and S. Patel, "SOBER cryptanalysis", Pre-proceedings of Fast Software Encryption '99, 1999, pp. 303-314.

[7]. G. Rose, "S32: A Fast Stream Cipher based on Linear Feedback over GF(2^{32})".Unpublished report, QUALCOMM Australia, Suite 410 Birkenhead Point, Drummoyne NSW 2047 Australia, 1998. See http://www.home.aone.net.au/qualcomm.

[8]. P. Hawkes and G. Rose, "Primitive Specification and Supporting Documentation for sober-t32 submission to NESSIE," *First NESSIE Workshop, Proceedings* 2000.

[9]. S. Babbage and J. Lano, "Probabilistic Factors in the Sober-t Stream Ciphers," third open NESSIE Workshop, proceeding, 2002

[10]. Wang Zhiyaun, Huang Jianhua and Guan Ziming, "The SOBER Family Ciphers Reconfigurable Processing Architecture Design," *fifth internation conference on Information assurance Security*, 2009

[11] P. Ekdahl and T. Johansson, "Snow". Primitive submitted to NESSIE, Sep.2000.

[12]. Patraik Ekdahl and Thomoas Johansson, "Distinguishing attacks on SOBER-t16 and SOBER-t32," *Fast Software Encrytion*, FSE 2002, Springer-Verlag, LNCS 2365, pp. 210-224

[13] S. Babbage, C. De Cannière, J. Lano, B. Preneel and J. Vandewalle, "Cryptanalysis of SOBER-t32", Pre-proceedings of Fast Software Encryption FSE2003, February 1999, pp. 119-136.

[14]Tore Herlestam "On functions of Linear Shift Register Sequence," Eurocrypt 85, LNCS 219, F. Pichler, Ed., Springer-Verlag, pp.119-129, 1985.

[15]Christopher De Canniere, "Guess and Determine Attack on SOBER", NES/DOC/KUL/WP5/010/a

[16]Dai Watanable and soichi Furuya, "A MAC forgery attack on SOBER-128", Proc. Fast Software Encryption 2004, Springer 2004

[17] P. Hawkes and G. Rose, "Turing: a fast steam cipher," First Software Encryption, FSE 2003, pre-proceedings, pp. 307-324, 2003.

[18]. G. Rose, "A Stream Cipher based on Linear Feedback over GF(2^8)", in C. Boyd, Editor, *Proc. Australian Conference on Information Security and Privacy*, Springer-Verlag 1998.

[19]. J. Hastad and M. Naslund Bmgl: "Synchronous key-stream generator with provable security". Primitive submitted to *NESSIE*, Sep.2000.

[20]JOO Yeon Cho and Josef Pieprzyk, "Algebraic Attacks on SOBER –t 32 and SOBER-t16 without stuttering". Fast Software Encryption-FSE 2004

[21]M U Bokhari and Faheem Masoodi, "Comparative Analysis of Structures and Attacks on Various Stream Ciphers". Proceedings of the 4th National Conference; INDIACom-2010

[22] M. Dichtl and M. Schafheutle, "Linearity Properties of the SOBER-t32 Key Loading", NESSIE Public Document NES/DOC/SAG/WP5/046/1, November 2001.

[23].NESSIE: New European Schemes for Signatures, Integrity, and Encryption. See http://www.cryptonessie.org.

[24]. Philip Hawkes, Cameron McDonald, Michael Paddon, Gregory G.Rose, and Miriam Wiggers de Vries, "Primitive Specifications for NLSv2"

[25]. P. Topark and P. Kanivichaporn, "SOBER-t-16 and SOBER-t-32," http://citeseerx.ist.psu.edu/viewdoc.